Institut
Teknologi
Sepuluh Nopember

TUGAS AKHIR – KS141501

OTOMATISASI *TRADING CRYPTOCURRENCY* BERBASIS WEB MENGGUNAKAN METODE *SIMPLE MOVING AVERAGE* (STUDI KASUS: XLM/BTC PADA BITCOIN.CO.ID)

AUTOMATION OF CRYPTOCURRENCY TRADING WEB BASED USING SIMPLE MOVING AVERAGE METHOD (CASE STUDY: XLM / BTC BITCOIN.CO.ID)

FEDDY ANUGERAH PRATAMA
NRP 5211 100 028

Dosen Pembimbing
Faizal Johan Atletiko S.Kom., M.T.

Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KS141501

OTOMATISASI *TRADING CRYPTOCURRENCY* BERBASIS WEB MENGGUNAKAN METODE *SIMPLE MOVING AVERAGE* (STUDI KASUS XLM/BTC PADA BITCOIN.CO.ID)

Feddy Anugerah Pratama
NRP 05211140000028

Dosen Pembimbing
Faizal Johan Atletiko, S.Kom, M.T.

Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – KS141501

AUTOMATION OF CRYPTOCURRENCY TRADING WEB BASED USING SIMPLE MOVING AVERAGE METHOD (CASE STUDY XLM / BTC BITCOIN.CO.ID)

Feddy Anugerah Pratama
NRP 05211140000028

Supervisor
Faizal Johan Atletiko, S.Kom, M.T.

Departement of Information Systems
Faculty of Information Technology and Communication
Institut Teknologi Sepuluh Nopember
Surabaya 2018

LEMBAR PENGESAHAN

OTOMATISASI *TRADING CRYPTOCURRENCY* BERBASIS WEB MENGGUNAKAN METODE *SIMPLE MOVING AVERAGE* (STUDI KASUS XLM/BTC PADA BITCOIN.CO.ID)

TUGAS AKHIR
Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi dan Komunikasi
Institut Teknologi Sepuluh Nopember

Oleh:

FEDDY NUGERAH PRATAMA
NRP. 0521 11 4000 0028

Surabaya, 25 Juli 2018

KEPALA
DEPARTEMEN SISTEM INFORMASI

Dr. Ir. Aris Tjahyanto, M.Kom,
NIP 19650310 199102 1 001

LEMBAR PERSETUJUAN

OTOMATISASI *TRADING CRYPTOCURRENCY* BERBASIS WEB MENGGUNAKAN METODE *SIMPLE MOVING AVERAGE* (STUDI KASUS XLM/BTC PADA BITCOIN.CO.ID)

Disusun Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Departemen Sistem Informasi
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:

FEDDY ANUGERAH PRATAMA
NRP. 0521 11 4000 0028

Disetujui Tim Penguji: Tanggal Ujian: 20 Juli 2018
Periode Wisuda: September 2018

Faizal Johan Atletiko, S.Kom., M.T. **(Pembimbing I)**

Renny Pradina K., S.T., M.T., SCJP **(Penguji I)**

Radityo Prasetianto W., S.Kom., M.Kom. **(Penguji II)**

OTOMATISASI *TRADING CRYPTOCURRENCY* BERBASIS WEB MENGGUNAKAN METODE *SIMPLE MOVING AVERAGE* (STUDI KASUS XLM/BTC PADA BITCOIN.CO.ID)

Nama Mahasiswa : Feddy Anugerah Pratama
NRP : 5211100028
Jurusan : Sistem Informasi FTIF-ITS
Pembimbing 1 : Faizal Johan Atletiko, S.Kom, M.T.

ABSTRAK

Saat ini bitcoin menjadi salah satu mata uang atau komoditas di dalam dunia maya yang dapat digunakan untuk membeli suatu barang. Bitcoin.co.id adalah sebuah situs agar dapat menukarkan rupiah dengan bitcoin. Dalam bitcoin.co.id telah disediakan API untuk kita agar dapat bertransaksi secara otomatis untuk bertukar uang elektronik yang disebut cryptocurrency yaitu dengan menggunakan aplikasi kita sendiri. Dengan adanya bitcoin.co.id banyak orang yang memiliki tujuan untuk memperoleh keuntungan dalam pertukaran rupiah dengan bitcoin. Namun, karena harga bitcoin yang berfluktuasi sangat cepat, sangat sulit sekali untuk menghitung harga sebenarnya dari bitcoin.

Simple moving average adalah salah satu metode yang umum digunakan untuk menghitung rata-rata harga dalam periode tertentu. Hasil dari penelitian ini adalah sebuah aplikasi berbasis web yang dapat menampilkan harga moving average serta melakukan pengambilan keputusan untuk menjual dan membeli secara otomatis menggunakan metode simple moving average. Pengambilan keputusan untuk perhitungan dan untuk jual-beli dilakukan setiap 10 detik sekali menggunakan javascript. Setelah aplikasi berjalan selama satu bulan, dapat diperoleh hasil dari jual-beli menggunakan metode simple moving average yang membuktikan bahwa aplikasi yang

menggunakan metode simple moving average mengalami kerugian.,

Kata Kunci: Simple Moving Average, Cryptocurrency, Bitcoin, API.

AUTOMATION OF CRYPTOCURRENCY TRADING WEB BASED USING SIMPLE MOVING AVERAGE METHOD (CASE STUDY XLM / BTC BITCOIN.CO.ID)

Student Name : Feddy Anugerah Pratama
NRP : 5211100028
Department : Sistem Informasi FTIF-ITS
Supervisor Lecturer 1 : Faizal Johan Atletiko, S.Kom, M.T

ABSTRACT

Bitcoin be one currency or commodity in the virtual world that can be used to buy an item. Bitcoin.co.id is a website to exchange rupiah with bitcoin. In bitcoin.co.id has provided API for us to be able to transact automatically to exchange electronic money called cryptocurrency that is by using our own application. With the existence of bitcoin.co.id many people have a goal to gain profit in exchange of rupiah with bitcoin. However, because the price of fluctuating bitcoins is very fast, it is very difficult to calculate the actual price of bitcoin.

Simple moving averages are one common method used to calculate the average price in a given period. The result of this research is a web-based application that can display moving average price and make decision to sell and buy automatically using simple moving average method. Decision-making for calculation and for buy and sell is done every 10 seconds once using javascript. After the application runs for one month, can be obtained from the buy and sell using simple moving average method that proves that applications using the method of moving average incur losses

Keywords: Simple Moving Average, Cryptocurrency, Bitcoin, E-Cash, API, Bot.

KATA PENGANTAR

Syukur Alhamdulillah terucap atas segala petunjuk, pertolongan, rahmat dan kekuatan yang diberikan oleh Allah SWT kepada penulis sehingga dapat menyelesaikan buku tugas akhir dengan judul:

Otomatisasi *Trading Cryptocurrency* Berbasis Web Menggunakan Metode *Simple Moving Average* (Studi Kasus Xlm/Btc Pada Bitcoin.co.id)

Pada kesempatan ini, penulis ingin menyampaikan terima kasih kepada semua pihak yang telah memberikan dukungan, bimbingan, arahan, bantuan, dan semangat dalam menyelesaikan tugas akhir ini, yaitu:

- Orang tua yang senantiasa mendoakan dan memberikan semangat, dan adik tercinta yang selalu mendorong untuk menyelesaikan tugas akhir tepat waktu.
- Bapak Faizal Johan Atletiko, S.Kom, M.T selaku dosen pembimbing yang yang telah meluangkan waktu untuk membimbing, mengarahkan dan mendukung dalam penyelesaian tugas akhir.
- Ibu Renny Pradina, S.T., M.T., selaku dosen wali yang senantiasa memberikan pengarahan selama penulis menempuh masa perkuliahan dan pengerjaan tugas akhir.

- Teman-teman Lab ADDI, Rachmat Arif, Andris, Brantas, Hendra, Ogi, Idam, dan Hufadz yang telah memberikan waktu untuk berdiskusi dan saling memberikan semangat dalam menyelesaikan tugas akhir.

Penyusunan tugas akhir ini masih jauh dari sempurna, untuk itu penulis menerima kritik dan saran yang membangun untuk perbaikan di masa mendatang. Semoga tugas akhir ini dapat menjadi salah satu acuan bagi penelitian-penelitian yang serupa dan bermanfaat bagi pembaca.

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

Pada bab pendahuluan akan diuraikan proses indentifikasi masalah meliputi latar belakang masalah, perumusan masalah, batasan masalah, tujuan dan manfaat dalam penelitian tugas akhir.

1.1.Latar Belakang

E-cash adalah sebuah sistem pembayaran menggunakan uang elektronik yang diperkenalkan oleh David Chaum pada tahun 1983 [2]. *E-cash* bertujuan untuk membayar tagihan tanpa membuka privasi dari si pembayar, sehingga bisa meminimalir kejahatan oleh pihak ketiga. Ketika membayar tagihan, pihak ketiga dapat dengan mudah menentukan gaya hidup si pembayar tagihan dengan tidak adanya *E-cash* [2].
E-cash memiliki kelemahan yaitu dapat digunakan untuk kegiatan kriminal berupa penipuan [3]. Karena kelemahan itu maka adanya pihak penengah diperlukan yaitu Escrow. Escrow adalah pihak ketiga yang ditunjuk untuk perantara menyimpan aset sementara yang akan ditransaksikan dan harus terpercaya dan bersifat netral.

Bitcoin adalah salah satu bentuk perkembangan e-cash yang memungkinkan untuk mentransfer uang secara peer-to-peer tanpa ada institusi finansial yang mengatur, sehingga fee untuk pertukaran tersebut sangat rendah. Bitcoin bisa digunakan untuk membayar barang yang kita beli dan juga untuk membayar jasa.

Di Indonesia, salah satu pihak netral untuk tempat pertukaran mata uang elektronik Bitcoin yaitu Bitcoin.co.id. Telah disediakan juga API untuk mengotomatisasi pembayaran agar bitcoin yang kita terima otomatis menjadi kurs rupiah. Dengan adanya API tersebut, kita bisa mengembangkan suatu aplikasi yang memungkinkan kita untuk melakukan perdagangan secara otomatis.

Pembuatan dokumen tugas akhir ini bertujuan untuk mendokumentasikan hasil dari implementasi bot menggunakan metode simple moving average untuk membuktikan apakah metode moving average dapat menguntungkan atau tidak.

1.2. Perumusan Masalah

Berdasarkan uraian latar belakang di atas, maka rumusan permasalahan yang menjadi fokus dalam penelitian tugas akhir ini adalah:

1. Bagaimana cara mengakuisisi data transaksi jual beli pada Bitcoin.co.id?
2. Bagaimana sistem mengetahui kapan waktunya membeli dan menjual?
3. Bagaimana cara sistem melakukan perhitungan menggunakan *simple moving average*?
4. Bagaimana cara merancang *platform* yang mampu menampilkan visualisasi data transaksi yang infomatif terhadap user?

1.3. Batasan Masalah

Batasan masalah pada Tugas Akhir ini antara lain sebagai berikut:

1. Pemanfaatan framework Codeigniter untuk pembuatan bot.
2. Pair perdagangan Stellar/Bitcoin.
3. Menggunakan metode moving average 12 periode dan 20 periode, 1 periode 15 menit agar aplikasi hanya menunggu pemrosesan data transaksi tidak terlalu lama
4. Menggunakan modal sebanyak 95 koin Stellar
5. Bot hanya berjalan 1 bulan.

1.4.Tujuan

Berdasarkan hasil perumusan masalah dan batasan masalah yang telah disebutkan sebelumnya, maka tujuan yang akan dicapai dari tugas akhir ini adalah:

1. Mengetahui cara mengakuisisi data transaksi jual beli pada Bitcoin.co.id.
2. Sistem dapat mengetahui kapan waktunya membeli dan menjual
3. Mengetahui bagaimana cara sistem melakukan perhitungan menggunakan *simple moving average.*
4. Mengetahui cara merancang *platform* yang mampu menampilkan visualisasi data transaksi yang infomatif terhadap user

1.5.Manfaat

Manfaat yang didapatkan dari penelitian ini adalah sebagai berikut:

1. Dapat dijadikan referensi pembuatan *bot trading*.
2. Mengetahui hasil dari metode *simple moving average* pada implementasi sebenarnya.

1.6.Relevansi

Tugas akhir ini berkaitan dengan mata kuliah Pemrograman Berbasis Web, Analisa dan Desain Perangkat Lunak dan Konstruksi Pengembangan Perangkat Lunak, Penggalian Data Analitika Bisnis.

“Halaman ini sengaja dikosongkan”

BAB II
TINJAUAN PUSTAKA

Bab ini akan menjelaskan mengenai penelitian sebelumnya dan dasar teori yang dijadikan acuan atau landasan dalam pengerjaan tugas akhir ini.

2.1 Studi Sebelumnya

Pada subbab ini dijelaskan tentang referensi penelitian yang berkaitan dengan tugas akhir. Pada bagian ini memaparkan acuan penelitian sebelumnya yang digunakan oleh penulis dalam melakukan penelitiannya.

1. Penelitian pertama mengenai *Pembuatan Market Expert Advisor pada Currency Market menggunakan Fibonacci, Stochastic dan MACD Indicator.* Penulis meneliti tentang bagaimana pembuatan dari aplikasi yang memberikan sinyal jual dan beli [4].
2. Penelitian kedua mengenai *Perancangan dan Implementasi Prosessor Scrypt Untuk Cryptocurrency Dengan Arsitektur Pipeline Berbasis FPGA* meneliti tentang bagaimana pembuatan prosessor Scrypt untuk Cryptocurrency yang digunakan untuk kegiatan yang disebut mining [5].

2.2 Dasar Teori

Bagian ini akan menjelaskan mengenai konsep atau teori yang berkaitan dengan tugas akhir.

2.2.1 E-Cash

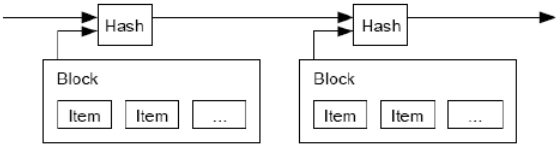
E-Cash merupakan saldo yang tertera pada aplikasi dan sebanding dengan jumlah yang disetorkan pada intitusi finansial [2]. Saldo tersebut bisa digunakan untuk membeli barang, membayar tagihan, pajak dan sebagainya. Namun di sisi lain E-cash memiliki kelemahan yaitu pihak institusi finansial dapat melihat kebiasaan kita membayar tagihan, sehingga privasi pelanggan menjadi dipertanyakan [2]. Pada tahun 1983, David Chaum memunculkan sebuah ide yang membuat sebuah pembayaran tagihan yang tidak dapat ditelusuri tetapi pembayaran itu telah divalidasi dengan benar. Ide yang di usulkan oleh David Chaum tersebut adalah seperti sebuah ballot dimasukkan ke amplop di dalam pemilihan yang diberi tanda dan memiliki kopi karbon. Untuk mengetahui apakah valid atau tidaknya sebuah amplop yang disegel tersebut, yaitu menggunakan amplop khusus sehingga pemilih mengetahui bahwa amplop miliknya telah dihitung dan yang lainnya tidak tahu milik siapa itu.

2.2.2 Bitcoin

Bitcoin adalah versi murni peer-to-peer electronic cash yang memperbolehkan pembayaran secara online langsung dikirimkan kepada dari seseorang ke orang lain tanpa melalui institusi finansial [1]. Tanda digital digunakan untuk menyelesaikan masalah tersebut, tapi masih diperlukan pihak ketiga untuk mencegah pembayaran ganda(double-spending). Bitcoin memberikan solusi untuk menyelesaikan masalah dari pembayaran ganda tersebut menggunakan jaringan peer-to-peer.

Dengan melakukan hashing pada waktu transaksi pada jaringan dalam rantai hash-based proof-of-work. Pembayaran melalui internet membutuhkan pihak ketiga yang terpercaya yang memproses pembayaran elektronik. Karena transaksi bisa dibatalkan, maka pihak ketiga harus benar-benar hati-hati

terhadap pelanggannya. Persentase dari penipuan pun dianggap dapat diterima dan tidak bisa dihindari. Kerugian ini bisa dibebankan pada pelanggan secara langsung. Solusi yang diusulkan dimulai dengan pencatatan timestamp ke dalam block. Di dalam timestamp terdapat timestamp sebelumnya pada hash tersebut membentuk sebuah rantai. Berikut ini adalah gambar untuk memperjelasnya:



Gambar 2.1 *Timestamp Chain* [1].

Di dalam sistem bitcoin terdapat metode proof-of-work untuk memvalidasi transaksi yang sudah di hash di dalam blok yang biasa disebut mining. Proses ini berjalan dengan melakukan hashing untuk memvalidasi hash tersebut. Metode proof-of-work memecahkan masalah dalam pengambilan keputusan bersama. Jika yang terbanyak adalah berdasarkan one-ip-address-one-vote maka akan dikalahkan oleh seseorang yang memiliki banyak IP. Maka dari itu proof-of-work adalah one-CPU-one-vote. Keputusan terbanyak direpresentasikan dengan rantai terpanjang. Untuk mengkompensasi kenaikan kecepatan hardware dan bermacam macam node yang berjalan, kesulitan dari proof-of-work akan disesuaikan berdasarkan rata-rata blok per jam. Jika mereka menyelesaikan terlalu cepat maka kesulitan akan bertambah.

Berikut ini bagaimana jaringan bitcoin bekerja:

1. Transaksi yang baru akan di broadcast kepada semua node.

2. Setiap node memasukkan transaksi baru ke dalam block.

3. Setiap node bekerja untuk mencari tingkat kesulitan block tersebut

4. Ketika node menemukan proof-of-work, node tersebut akan membroadcast block tersebut kepada semua node

5. Node menerima block apabila hanya jika semua transaksi di dalamnya valid dan tidak sudah digunakan.

Node mengekspresikan penerimaan mereka atas blok tersebut dengan mengerjakan pembuatan blok berikutnya di rantai, menggunakan hash dari blok yang diterima sebagai hash sebelumnya.

2.2.3 Stellar/Lumen

Stellar adalah organisasi non-profit yang didirikan oleh Jed McCaleb dan Joyce Kim. Pada tahun 2014 mereka mendirikan Stellar Development Foundation. Awalnya Stellar adalah sebuah proyek koin yang menggunakan Ripple Protokol. Setelah melakukan beberapa perubahan, Stellar membuat versi terbaru dengan algoritma baru dirilis tahun 2015. Pada tahun 2017 IBM dan KlickEx mengumumkan akan menjadi pendukung stellar.

Saat ini Stellar menjadi sebuah protokol open-source untuk pertukaran uang. Tiap server menyimpan catatan semua akun pada jaringannya. Catatan ini disimpan ke dalam database yang disebut “Ledger”. Tiap server menawarkan perubahan ledger dengan menawarkan transaksi yang mana mengubah saldo akun menjadi saldo akun lainnya [6].

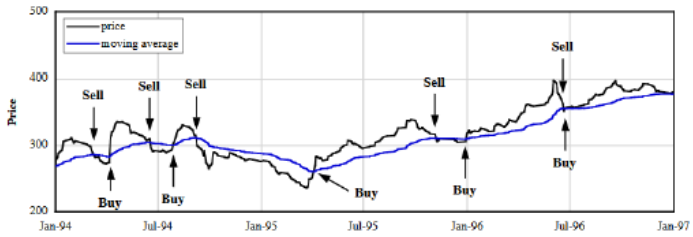
Saat ini Stellar/STR diganti namanya menjadi Lumen/XLM agar sesuai dengan standar karena mengikuti Federation Byzantine Agreement. [7]

2.2.4 Moving Average

Sistem trading adalah sebuah metode sistematis untuk menjual dan membeli intrumen keuangan dengan secara konsisten untuk mendapatkan uang [8]. Teknikal analisis adalah sebuah pendekatan untuk memprediksi pergerakan harga akan datang berdasarkan pola dan volume dari harga tersebut. Teknikal analisis biasanya dilakukan dengan cara mencatat aktivitas dari jual beli di dalam sebuah grafik.

Moving average adalah salah satu metode dalam jual beli yaitu dengan menghitung rata-rata dari penjualan dan pembelian selama periode tertentu [8]. Pergerakan harga dapat keatas dan ke bawah yang biasa disebut uptrend dan downtrend. Uptrend didefinisikan sebagai periode kenaikan harga dan downtrend didefinisikan sebagai periode penurunan harga. Ketika harga naik dan melewati di atas harga rata-rata maka akan diidentifikasi sebagai uptrend dan merupakan waktu terbaik untuk membeli. Ketika harga melewati di bawah rata-rata maka akan ditentukan sebagai downtrend dan merupakan waktu terbaik untuk menjual.

Berikut ini adalah gambar untuk lebih memahami kapan waktu membeli dan menjual.



Gambar 2.2 Buy and Sell Moving Average [8]

Berikut ini adalah contoh perhitungan moving average:

Contoh dari 3 periode Simple Moving Average (SMA) Jumlah Nilai Periode / Jumlah Periode Harga yang akan digunakan: 5, 6, 7, 8, 9 Hari Pertama Periode 3 SMA: (5 + 6 + 7) / 3 = 6 Hari

Kedua dari 3 Periode SMA: (6 + 7 + 8) / 3 = 7 Hari Ketiga dari 3 Periode SMA: (7 + 8 + 9) / 3 = 8 Untuk periode Moving Average bisa disesuaikan sesuai yang dinginkan yaitu periode bulanan, harian, perjam, dll.

BAB III
METODOLOGI PENELITIAN

Metode penelitian yang digunakan untuk menyusun tugas aakhir ini adalah metode *waterfall*. Metode *waterfall* merupakan salah satu metode yang dilakukan dalam pembuatan sebuah aplikasi.

3.1. Tahap Perancangan

Tahap awal dalam penelitian ini adalah tahapan perancangan. Tahap ini merupakan persiapan yang harus dilakukan sebelum melakukan *pembuatan*, yaitu:

a. Tahap Studi Literatur

Tahap studi literatur dilakukan dengan tujuan untuk dapat memahami konsep, metode, dan teknologi sesuai bahasan dan permasalahan sehingga dapat memberi solusi mengenai permasalahan yang akan digunakan dalam penyusunan tugas akhir.

3.2.Tahap Akuisisi Data

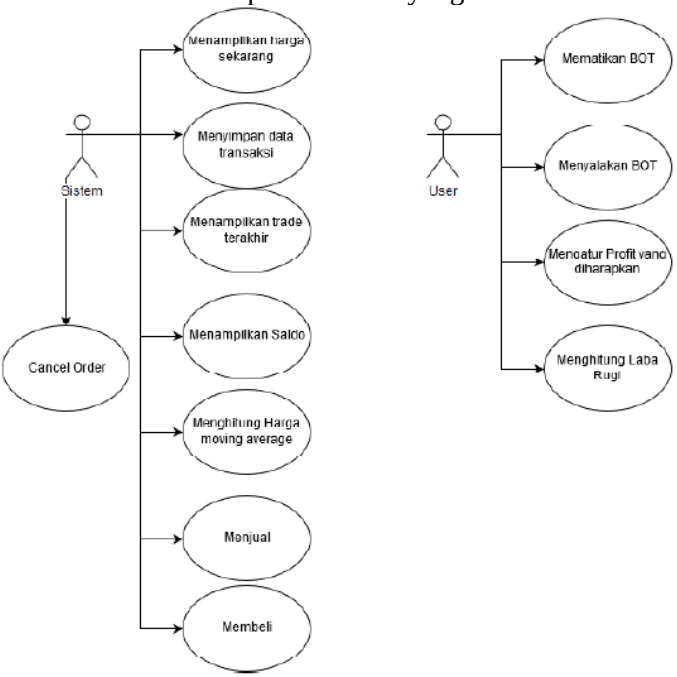
Proses akuisisi data akan dilakukan secara otomatis setiap interval 10 detik dengan javascript karena server hanya memperbolehkan 3 *request* dalam satu detik dan dengan memperkirakan waktu respon maksimal sebanyak 2 detik. Data yang diperoleh berasal dari API yang disediakan oleh layanan jual-beli yaitu Bitcoin.co.id. Data yang akan diambil adalah data transaksi jual-beli dan data yang berkaitan dengan pengguna seperti informasi saldo, lalu informasi order, dll.

3.3.Pembuatan User Interface

Data yang berasal dari API bitcoin.co.id akan di visualisasikan agar user lebih memahami dan tampilan akan memiliki sifat informatif.

3.4. Pembuatan Bot dan Fungsi Pengaturan lainnya

Pembuatan bot akan dilakukan dengan menggunakan bahasa pemrograman PHP dan menggunakan framework Codeigniter. Berikut ini adalah fitur pada website yang akan dibuat.



Gambar 3.1 Usecase Sistem dan User

Fitur-fitur yang akan di buat pada sistem adalah:

1. Menampilkan Harga sekarang.
2. Menyimpan data transaksi jual beli.
3. Menampilkan trade terakhir.
4. Menampilkan Saldo.
5. Menghitung nilai *moving average* dari harga.
6. Menjual.
7. Membeli.

Fitur-fitur yang akan dibuat untuk pengguna adalah:

1. Menyalakan Bot
2. Mematikan Bot
3. Mengatur Profit yang diharapkan.
4. Menghitung rugi laba.

3.5. Perancangan dan Pembuatan Visualisasi

Pembuatan visualisasi dari bot dengan grafik akan dilakukan menggunakan library dari HighStock.js yang memungkinkan untuk membuat visualisasi dengan model candlestick yang biasa digunakan pada *platform* jual-beli lainnya. Visualisasi grafik terbatas hanya menampilkan *candlestick* dengan ditambah *marker* atau penanda dimana waktu sistem membeli dan menjual. Untuk visualisasi bentuk grafik dari moving average tidak tersedia karena pada Bitcoin.co.id sudah tersedia fitur visualisasi dengan metode *simple moving average*. Visualisasi *simple moving average* hanya berupa angka yaitu menampilkan *simple moving average* saat ini dan diperbarui setiap 10 detik bersamaan dengan proses akuisisi data yang telah dijelaskan sebelumnya.

3.6. Pengujian Aplikasi

Pengujian aplikasi dilakukan dengan menguji fungsional dari aplikasi apakah sudah berjalan sesuai yang diharapkan atau tidak dengan menggunakan *test case*. Lalu untuk non fungsional yaitu dengan membandingkan bot berjalan menggunakan 12 periode dengan 20 periode masing masing selama 15 hari.

BAB IV
PERANCANGAN

Pada bab ini akan dijelaskan bagaimana cara memperoleh data yang akan digunakan untuk tahap implementasi.

4.1. Perancangan Pengumpulan Data Bitcoin.co.id

Pada tahap ini akan dijelaskan cara untuk memperoleh data yang diperlukan dari Bitcoin.co.id.

Bitcoin.co.id telah menyiapkan API untuk user agar user bisa memperoleh data yang diperlukan untuk keperluan pembuatan aplikasi yang menggunakan algoritma perhitungan pengguna sendiri.

Berikut adalah daftar dari fungsi yang dipanggil untuk mendapatkan data:

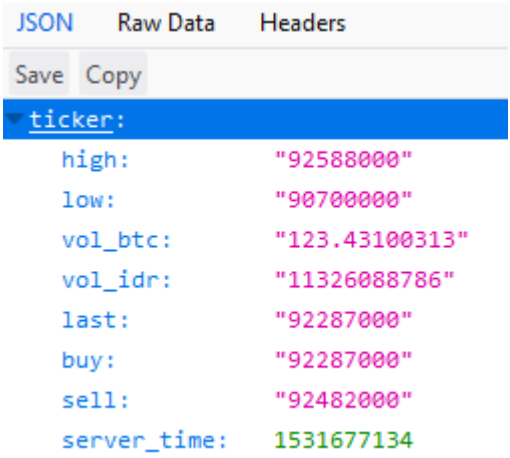
1. Public API

   Public API adalah sebuah API yang tidak diharuskan untuk memasukkan API key untuk mengakses sebuah fungsi. Lihat gambar 4.1.

   Ticker BTC/IDR - https://indodax.com/api/btc_idr/ticker
   Trades BTC/IDR - https://indodax.com/api/btc_idr/trades
   Depth BTC/IDR - https://indodax.com/api/btc_idr/depth

   Gambar 4.1 *Public API*

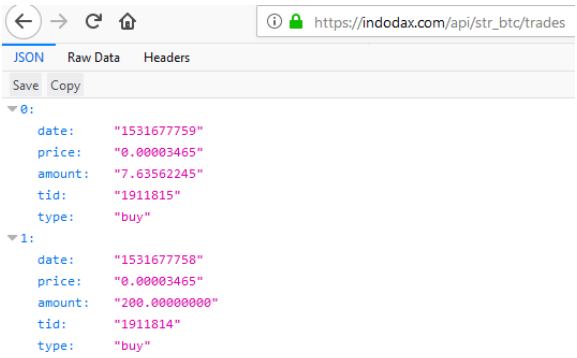
   Ticker adalah data perubahan suatu pasar. Berikut adalah hasil dari memanggil fungsi tersebut. Lihat gambar 4.2.



Gambar 4.2 Response Ticker API

Pada ticker kita bisa melihat harga tertinggi dari pair BTC_IDR yaitu Rp 92.588.000 rupiah untuk 1 bitcoin, harga terendahnya Rp 90.700.000 rupiah untuk 1 bitcoin. Lalu volume transaksi bitcoin perhari adalah 123 bitcoin. Sedangkan volume rupiah yang digunakan untuk transaksi bitcoin perhari adalah Rp 11.326.088.786. Kita juga bisa melihat harga terakhir serta harga beli dan jual saat ini. Lalu yang terakhir kita bisa melihat waktu pada server yang ditampilkan dengan format Unix Timestamp.

Kita juga bisa melihat data transaksi jual-beli. Dengan mengakses metode trades. Berikut hasilnya dapat dilihat pada gambar 4.3:



Gambar 4.3 Response Trade API

Dapat dilihat data transaksi ketika mengakses fungsi tersebut.

Yang terakhir adalah Depth untuk mengetahui seberapa banyak pengguna yang melakukan open order jual atau beli.

2. Private API

Untuk menggunakan private API kita memerlukan API-Key dan secret key untuk bisa bertransaksi dengan akun kita.

Ada 3 hak akses yang dapat digunakan.

1. View : Berfungsi untuk memperoleh data.

2. Trade : Melakukan jual beli.

3. Withdraw : Mengambil asset.

Tabel 4.1 Fungsi Pada Private API

View	getInfo, transHistory, tradeHistory, openOrders, orderHistory, getOrder
Trade	trade, cancelOrder
Withdraw	withdrawCoin

Hak akses view adalah berfungsi untuk memperoleh data yang kita perlukan.

Hak akses trade adalah berfungsi untuk melakukan aksi yaitu order atau cancel order.

Hak akses Withdraw adalah berfungsi untuk menarik asset kita yaitu koin yang akan dipidahkan ke tempat lain.

Berikut ini adalah penjelasan masing masing fungsi pada hak akses view:

Tabel 4.2 Penjelasan Kategori View API

View	Fungsi
getInfo	Menampilkan jumlah asset dari user
transHistory	Menampilkan daftar dari deposit dan withdraw user

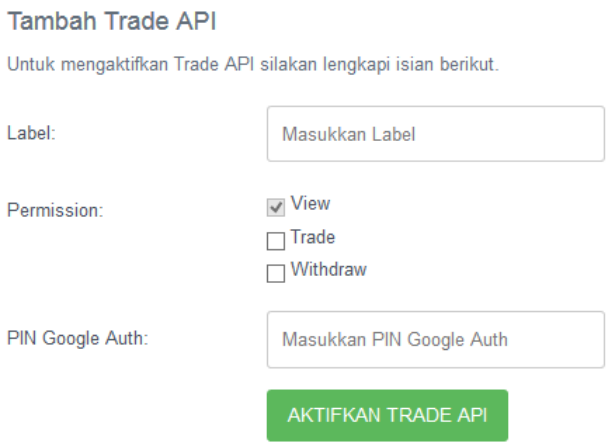
tradeHistory	Menampilkan informasi tentang riwayat dari penjualan dan pembelian
openOrder	Menampikan daftar dari order jual atau beli yang saat ini belum terpenuhi.
orderHistory	Menampilkan riwayat dari order jual atau beli.
getOrder	Menampilkan detail dari order tertentu

Berikut ini adalah fungsi dari metode yang ada pada hak akses trade:

Tabel 4.3 Penjelasan Kategori Trade API

Trade	Fungsi
trade	Untuk melakukan order baru
cancelOrder	Untuk membatalkan order secara spesifik

Untuk mengakses *private API* dibutuhkan API key dan secret key. Untuk mendapatkan API key, maka harus mengisi form berikut ini pada bitcoin.co.id. Lihat gambar 4.4.



Gambar 4.4 Form Tambah Trade API

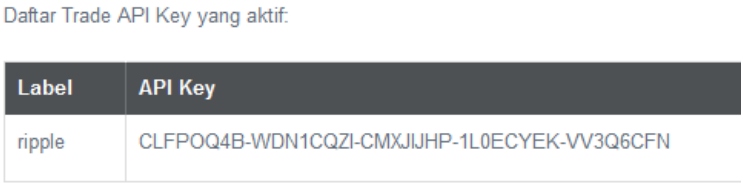
Pada form tambah trade API, label adalah nama trade api. Lalu pada permission dapat dilihat bahwa hanya memiliki 3 permission.

Permission view berkaitan dengan pengambilan data yang sifatnya hanya untuk dilihat dan diproses lebih lanjut untuk keperluan pengembangan algoritma sendiri.

Permission trade berkaitan dengan aktifitas pengambilan keputusan untuk jual beli.

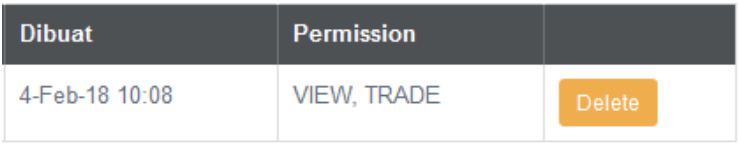
Permission withdraw berkaitan dengan pemindahan aset ke tempat lain.

Setelah mengaktifkan trade API maka akan terlihat hasil seperti berikut ini pada gambar 4.5:



Gambar 4.5 API Key

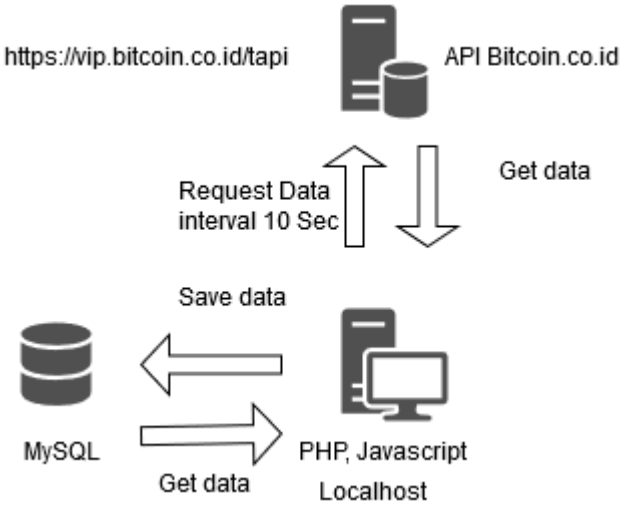
Lalu dapat dilihat pula permission yang ada pada trade api tersebut. Berikut ini adalah hasilnya pada gambar 4.6:



Gambar 4.6 Permission API

4.2. Arsitektur Sistem

Pada subbab ini akan dijelaskan bagaimana arsitektur sistem yang akan dibuat. Berikut ini adalah gambar dari arsitektur sistem:



Gambar 4.7 Arsitektur Sistem

Pada gambar 4.7 arsitektur sistem terdari dari 3 bagian yaitu API server bitcoin.co.id, localhost, dan database MySQL. Localhost akan melakukan request data yang diperlukan setiap 10 detik sekali dengan menggunakan javascript. Kemudian data yang telah diperoleh akan disimpan ke dalam database MySQL. Penggunaan waktu 10 detik dimaksudkan agar waktu tersebut mencukupi dengan mengasumsikan waktu maksimal respon dari server adalah 2 detik dan setiap request perdetik memiliki batas 3 request. Sehingga akan ada minimal 15 request per 10 detik. Penggunaan interval 10 detik juga dimaksudkan agar pengambilan keputusan tidak terlalu lama mengingat cepatnya fluktuasi dari harga.

4.3. Hasil *Response Private API*

Setelah berhasil membuat trade API, pada subbab ini akan ditampilkan hasil dari tiap-tiap respon private API pada bitcoin.co.id.

4.3.1. Fungsi getInfo

Berikut ini adalah respon dari fungsi getInfo: Lihat gambar 4.7.

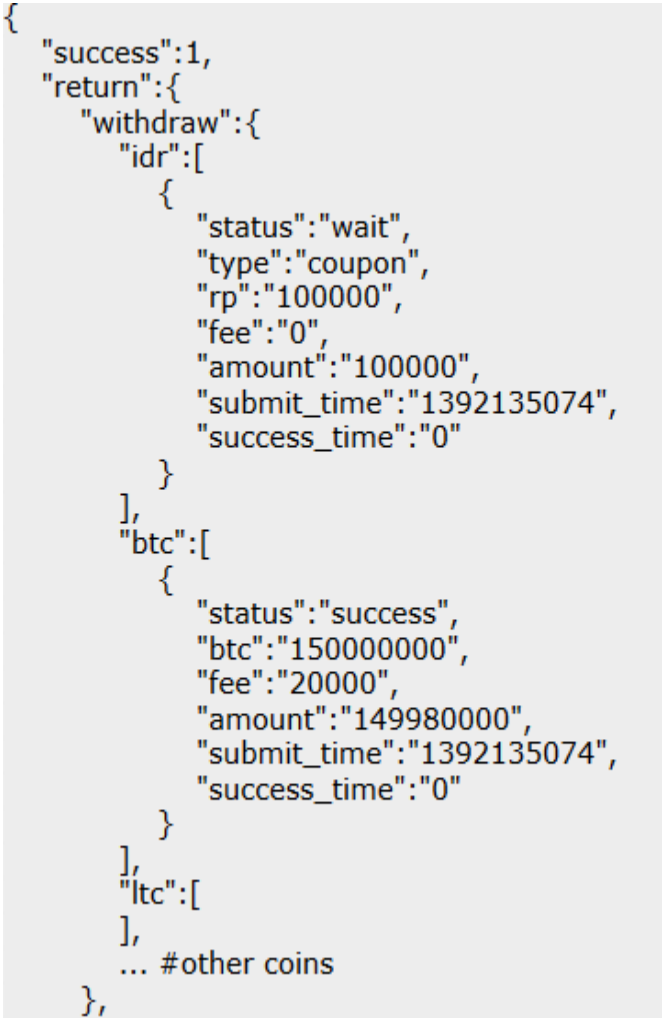

```
{
   "success":1,
   "return":{
      "balance":{
         "idr":2000000,
         "btc":1.52,
         "ltc":33.14,
         "doge":1600.299,
         "xpy":529.1239,
         ... #other coins balances
      },
      "server_time":1392225342
   }
}
```


Gambar 4.8 Response getInfo

Fungsi ini akan menampilkan semua saldo yang kita miliki pada akun bitcoin.co.id.

4.3.2. Fungsi transHistory

Berikut ini adalah respon dari fungsi transHistory: Lihat gambar 4.9 dan 4.10.



```
{
   "success":1,
   "return":{
      "withdraw":{
         "idr":[
            {
               "status":"wait",
               "type":"coupon",
               "rp":"100000",
               "fee":"0",
               "amount":"100000",
               "submit_time":"1392135074",
               "success_time":"0"
            }
         ],
         "btc":[
            {
               "status":"success",
               "btc":"150000000",
               "fee":"20000",
               "amount":"149980000",
               "submit_time":"1392135074",
               "success_time":"0"
            }
         ],
         "ltc":[
         ],
         ... #other coins
      },
```


Gambar 4.9 Response transHistory Withdraw

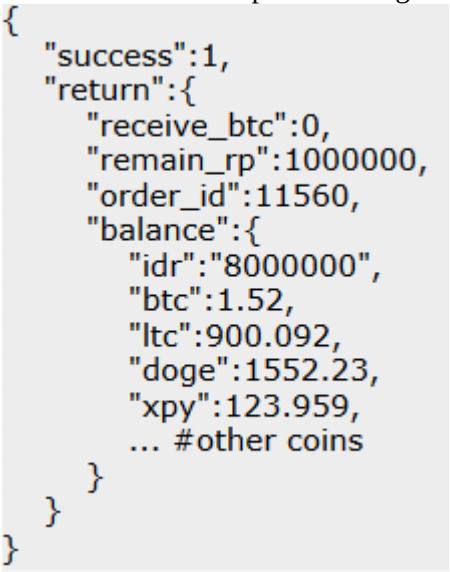
```
    "deposit":{
      "idr":[
        {
          "status":"success",
          "type":"bank",
          "rp":"10000000",
          "fee":"0",
          "amount":"10000000",
          "submit_time":"1392193569",
          "success_time":"1392193569"
        }
      ],
      "btc":[
        {
          "status":"success",
          "btc":"200000000",
          "amount":"200000000",
          "success_time":"1391979201"
        }
      ],
      ... #other coins
    }
  }
}
```


Gambar 4.10 Response transHistory Deposit

Pada gambar 4.9 adalah respon dari *withdraw*. Sedangkan gambar 4.10 adalah respon dari deposit.

4.3.3. Fungsi trade

Berikut ini adalah respon dari fungsi trade: Lihat gambar 4.11.



```
{
   "success":1,
   "return":{
      "receive_btc":0,
      "remain_rp":1000000,
      "order_id":11560,
      "balance":{
         "idr":"8000000",
         "btc":1.52,
         "ltc":900.092,
         "doge":1552.23,
         "xpy":123.959,
         ... #other coins
      }
   }
}
```


Gambar 4.11 Response Trade

Pada fungsi trade, diperlukan parameter lainnya untuk melakukan jual-beli secara spesifik. Berikut ini adalah parameter yang diperlukan untuk mengakses fungsi ini: Lihat gambar 4.12.

Parameter	Required?	Description	Value	Default
pair	Yes	Pair to get the information from	*btc_idr, ltc_btc, doge_btc, etc*	btc_idr
type	Yes	transaction type (buy or sell)	buy / sell	-
price	Yes	order price	numerical	-
idr	required on buying btc	amount of rupiah to buy btc	numerical	-
btc	required on selling btc	amount of btc to sell	numerical	-

Gambar 4.12 trade Parameter

4.3.4. Fungsi tradeHistory

Fungsi tradeHistory adalah fungsi untuk memberikan informasi riwayat transaksi jual beli.
Berikut ini adalah respon dari fungsi tradeHistory: Lihat gambar 4.13.

```
{
   "success":1,
   "return":{
      "trades":[
         {
            "trade_id":"2929",
            "order_id":"8123",
            "type":"buy",
            "btc":0.013,
            "price":"8068585",
            "fee":"1049",
            "trade_time":"1392226454"
         },
         {
            "trade_id":"2920",
            "order_id":"8111",
            "type":"sell",
            "btc":0.01499999,
            "price":"8086935",
            "fee":"1214",
            "trade_time":"1392225916"
         }
      ]
   }
}
```


Gambar 4.13 tradeHistory Response

Fungsi tradeHistory memiliki parameter yang harus dipenuhi untuk mengaksesnya. Berikut ini adalah parameter yang diperlukan untuk mengakses fungsi tradeHistory: Lihat gambar 4.14.

Parameter	Required?	Description	Value	Default
count	No	number of transaction which will be displayed	numerical	1000
from_id	No	first ID	numerical	0
end_id	No	end ID	numerical	∞
order	No	sort by	asc / desc	desc
since	No	start time		UNIX time
end	No	end time		UNIX time
pair	Yes	Pair to get the information from	*btc_idr, ltc_btc, doge_btc, etc*	btc_idr

Gambar 4.14 TradeHistory Parameter

4.3.5. Fungsi openOrder

Fungsi ini digunakan untuk melihat semuaa list open order saat ini. Berikut adalah respon dari fungsi openOrder: Lihat gambar 4.15.

```
{
   "success": 1,
   "return": {
      "orders": {
         "btc_idr": [
            {
               "order_id": "11567",
               "submit_time": "1392227908",
               "price": "10000000",
               "type": "buy",
               "order_idr": "1000000",
               "remain_idr": "1000000"
            }
         ],
         "ltc_btc": [
            {
               "order_id": "12345",
               "submit_time": "1392228122",
               "price": "8000000",
               "type": "sell",
               "order_ltc": "100000000",
               "remain_ltc": "100000000"
            }
         ]
      }
   }
}
```


Gambar 4.15 OpenOrder Response

4.3.6. Fungsi orderHistory

Fungsi ini digunakan untuk menampilkan riwayat dari order yang telah terpenuhi atau dibatalkan. Berikut ini adalah respon dari fungsi orderHistory Lihat gambar 4.16.

```
{
  "success": 1,
  "return": {
    "orders": [
      {
        "order_id": "11512",
        "type": "sell",
        "price": "5000000",
        "submit_time": "1392227908",
        "finish_time": "1392227978",
        "status": "filled",
        "order_btc": "0.00100000",
        "remain_btc": "0.00000000"
      },
      {
        "order_id": "11513",
        "type": "buy",
        "price": "5000000",
        "submit_time": "1392227908",
        "finish_time": "1392227978",
        "status": "cancelled",
        "order_idr": "1000",
        "remain_idr": "1000"
      }
    ]
  }
}
```


Gambar 4.16 OrderHistory Response

Untuk mengakses fungsi orderHistory, diperlukan parameter. Berikut ini adalah parameter yang diperlukan untuk mengakses fungsi orderHistory: Lihat gambar 4.17.

Parameter	Required?	Description	Value	Default
pair	Yes	Pair to get the information from	*btc_idr, ltc_btc, doge_btc, etc*	btc_idr
count	No		integer	100
from	No		integer	0

Gambar 4.17 OrderHistory Parameter

4.3.7. Fungsi getOrder

Fungsi getOrder adalah fungsi untuk menampilkan order secara spesifik. Berikut ini adalah respon dari fungsi getOrder: Lihat gambar 4.18.


```
{
   "success": 1,
   "return": {
      "order": {
         "order_id": "94425",
         "price": "0.00810000",
         "type": "sell",
         "order_ltc": "1.00000000",
         "remain_ltc": "0.53000000",
         "submit_time": "1497657065",
         "finish_time": "0",
         "status": "open"
      }
   }
}
```


Gambar 4.18 GetOrder Response

Fungsi getOrder memiliki parameter yang harus dipenuhi. Berikut ini adalah parameter yang diperlukan untuk mengakses fungsi getOrder: Lihat gambar 4.19.

Parameter	Required?	Description	Value	Default
pair	Yes	Pair to get the information from	*btc_idr, ltc_btc, doge_btc, etc*	btc_idr
order_id	Yes	Order ID	integer	-

Gambar 4.19 GetOrder Parameter

4.3.8. Fungsi cancelOrder

Fungsi ini adalah fungsi untuk membatalkan order aktif saat ini. Berikut ini adalah respon dari fungsi cancel order: Lihat gambar 4.20.

```
{
   "success":1,
   "return":{
      "order_id":11574,
      "type":"buy",
      "balance":{
         "idr":"5000000",
         "btc":2.5,
         "ltc":900.092,
         "doge":1552.23,
         "xpy":123.959,
         ... #other coins
      }
   }
}
```


Gambar 4.20 CancelOrder Response

Fungsi ini memerlukan parameter. Berikut ini adalah parameter yang diperlukan. Lihat gambar 4.21.

Parameter	Required?	Description	Value	Default
pair	Yes	Pair to get the information from	*btc_idr, ltc_btc, doge_btc, etc*	btc_idr
order_id	Yes	Order ID	numerical	-
type	Yes	transaction type (buy or sell)	buy / sell	-

Gambar 4.21 CancelOrder Parameter

4.3.9. Fungsi withdrawCoin

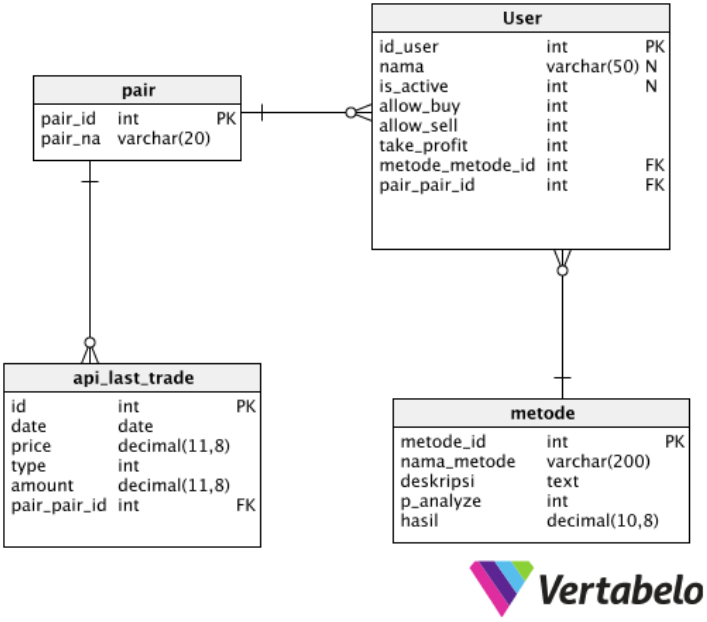
Fungsi ini digunakan untuk memindahkan aset ke tempat lain. Berikut ini adalah respon dari fungsi withdrawCoin: Lihat gambar 4.22.


```
{
   "success": 1,
   "status": "approved",
   "withdraw_currency": "xrp",
   "withdraw_address": "rwWr7KUZ3ZFwzgaDGjKBysADByzxvohQ3C",
   "withdraw_amount": "10000.00000000",
   "fee": "2.00000000",
   "amount_after_fee": "9998.00000000",
   "submit_time": "1509469200",
   "withdraw_id": "xrp-12345",
   "txid": "",
   "withdraw_memo": "123123"
}
```


Gambar 4.22 WithdrawCoin Response

4.4. Perancangan Database

Setelah memperoleh data dan informasi yang diperlukan tahap selanjutnya adalah merancang database.
Berikut ini adalah konseptual data model: Lihat gambar 4.23.



Gambar 4.23 ***Conceptual Data Model***

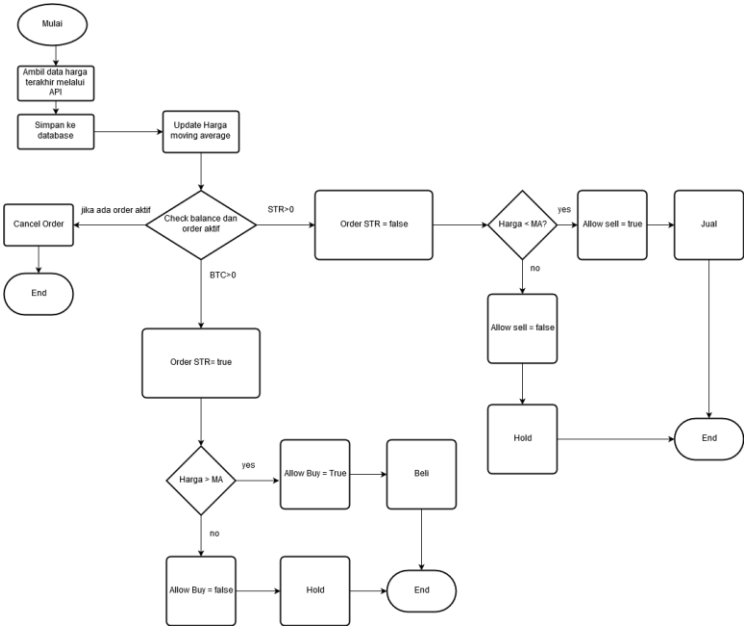
Ada 4 tabel akan dibuat yang diperlukan untuk menyimpan data.

Tabel 4.4 Database Tabel

Tabel	Fungsi
Pair	Menyimpan pair jual-beli
Api_last_trade	Menyimpan data jual-beli dari Bitcoin.co.id
User	Menyimpan list bot
Metode	Menyimpan metode yang digunakan

4.5. Perancangan FlowChart

Bab ini akan dijelaskan bagaimana alur dari aplikasi yang akan dibuat.
Berikut ini adalah flowchart dari aplikasi: Lihat gambar 4.24.



Gambar 4.24 Flowchart Sistem

Untuk melakukan order secara otomatis pastikan agar bot aktif terlebih dahulu. Sistem mengambil data harga terakhir dan menyimpan ke dalam database. Lalu melakukan perhitungan moving average dan menampilkannya.

Sistem akan melakukan cek apakah ada order aktif, jika ada order aktif sistem akan melakukan cancel pada semua order aktif yang belum terpenuhi. Jika tidak ada order aktif sistem

akan melakukan cek apakah kita memiliki saldo atau tidak. Apabila kita memiliki saldo, maka sistem akan melakukan penjualan atau pembelian

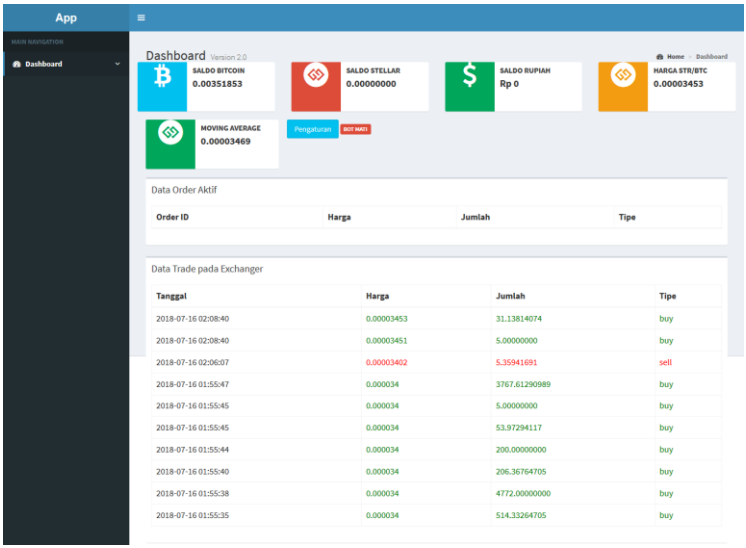
BAB V
IMPLEMENTASI

Bab ini menjelaskan proses pembuatan aplikasi. Pada aplikasi ini bot akan melakukan jual beli setiap 10 detik.

5.1. Pembuatan User Interface

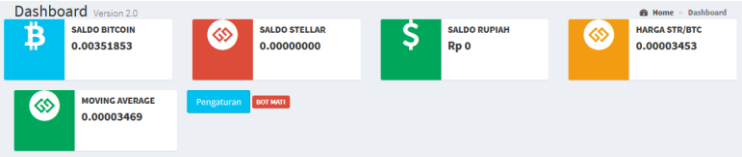
Pada sub bab ini akan dijelaskan mengenai pembuatan *user interface*.*User interface* dibuat menggunakan html,css, dan javascript.

Berikut ini adalah user interface aplikasi.



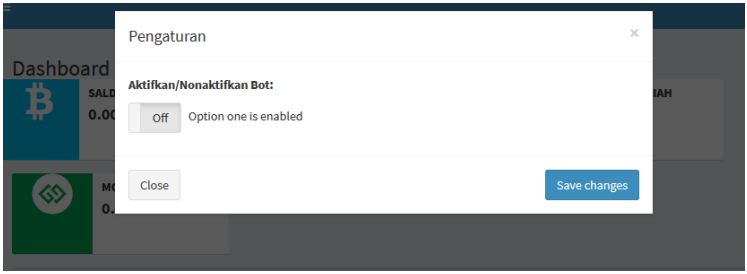
Gambar 5.1 Interface Aplikasi

Pada bagian atas adalah bagian yang berfungsi untuk menampilkan data saldo akun. Dapat dilihat pada gambar 5.1.



Gambar 5.2 Interface Tampilkan Saldo

Lalu menu pengaturan berfungsi untuk mengaktifkan dan menonaktifkan bot. Lihat gambar 5.2 dan 5.3.



Gambar 5.3 Interface Pengaturan

Untuk sisanya adalah tabel list data order aktif dan data 10 terakhir trade dari Bitcoin.co.id.

5.2. Pembuatan Bot dan Fungsi Pengaturan Lainnya

Pada bab ini akan dijelaskan bagaimana cara membuat bot dan fungsi pengaturan lainnya.

5.2.1. Fitur Menampilkan Balance

Untuk membuat fitur ini pertama-tama diperlukan untuk mengakses API dari Bitcoin.co.id.
Untuk mengakses API, akan dibuat fungsi khusus untuk menghubungkan server.
Membuat fungsi btcid_query($method, array $req = array()).
Berikut ini adalah kode programnya: Lihat gambar 5.4.

```
function btcid_query($method, array $req = array()) {
        // API settings
        $key = 'CLFPOQ4B-WDN1CQZI-CMXJIJHP-1L0ECYEK-VV3Q6CFN'; // your API-key
        $secret = 'bf5cd605b0ef4b7b0f870b6968f62730695462b3926c8ae4fcf6e443a47e933e7e1232564c4f0201';
        $req['method'] = $method;
        $req['nonce'] = time();
        // generate the POST data string
        $post_data = http_build_query($req, '', '&');
        $sign = hash_hmac('sha512', $post_data, $secret);
        // generate the extra headers
        $headers = array(
'Sign: '.$sign,
'Key: '.$key,
        );
        // our curl handle (initialize if required)
        static $ch = null;
        if (is_null($ch)) {
                $ch = curl_init();
                curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
                curl_setopt($ch, CURLOPT_USERAGENT, 'Mozilla/4.0 (compatible;
BITCOINCOID PHP client; '.php_uname('s').'; PHP/'.phpversion().')');
        }
        curl_setopt($ch, CURLOPT_URL, 'https://vip.bitcoin.co.id/tapi/');
        curl_setopt($ch, CURLOPT_POSTFIELDS, $post_data);
        curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);
        curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, FALSE);
        // run the query
        $res = curl_exec($ch);
        if ($res === false) throw new Exception('Could not get reply:
'.curl_error($ch));
        $dec = json_decode($res, true);
        if (!$dec) throw new Exception('Invalid data received, please make sure
connection is working and requested API exists: '.$res);
        curl_close($ch);
        $ch = null;
```

Gambar 5.4 Fungsi Query Bitcoin.co.id

Setelah itu membuat fungsi getBalance() yang akan dipanggil untuk memperoleh data saldo kita. Lihat gambar 5.5.

```
function getBalance(){
        $balance = array();

        $history = array('pair'=>'str_btc');
        $result = $this->api->btcid_query('getInfo');
        //$est = $this->api->getLaststridr();
        foreach ($result as $value) {
        $balance = $value;
}
        return $balance;
    }
```

Gambar 5.5 Fungsi Query Saldo

Pada fungsi getbalance(), akan dipanggil fungsi untuk memperoleh data dengan parameter metode getInfo.
Setelah itu tampilkan pada kotak. Berikut kode programnya:
Lihat gambar 5.6.

```
<div class="info-box-content">
  <span class="info-box-text">Saldo Stellar</span>
  <span class="info-box-number"><?php
  if (is_array($saldo) && isset($saldo)) {
  echo $saldo['balance']['str'];
  }
  else {
      echo "data tidak ditemukan";
  }
  ?></span>
</div>
<!-- /.info-box-content -->
```

Gambar 5.6 View Saldo

5.2.2. Fitur Menampilkan Harga

Pada subbab ini akan dijelaskan bagaimana cara untuk menampilkan data harga saat ini.
Pertama-tama buat fungsi untuk menegambil data harga terakhir pada Bitcoin.co.id
Berikut adalah kode programnya: Lihat gambar 5.7.

```
26      function getBalance(){
27          $balance = array();
28          $result = $this->api->btcid_query('getInfo');
29          foreach ($result as $value) {
30              $balance = $value;
31              }
32          return $balance;
33      }
```

Gambar 5.7 Fungsi Harga Terakhir

Setelah mengambil data, tampilkan data pada box.

5.2.3. Perhitungan *Moving Average*

Pada subbab ini akan dijelaskan bagaimana cara membuat fitur perhitungan *moving average.*

Untuk menghitung moving average pertama-tama kita membuat fungsi untuk mengambil data open price.

Berikut adalah kode programnya: Lihat gambar 5.8.

```
24      function openPrice($time1,$time2){
25          //gunakan sql between timestamp untuk menemukan data
26          //open = sort desc dan pilih top (1) harga
27          $date="date";
28          $this->db->select('price');
29          $this->db->from('api_last_trade');
30          $this->db->where("$date BETWEEN $time2 AND $time1");
31          $this->db->order_by('price', 'DESC');
32          $query = $this->db->get()->row('price');
33         if($query){
34             return $query;
35         }
36        else{
37            return ;
38        }
39      }
```

Gambar 5.8 Fungsi openPrice

Karena data yang diperoleh adalah data yang belum di klasifikasi, maka untuk membuat fungsi openprice() ,digunakan 2 timestamp. Yaitu time1 adalah now() dan time2 adalah now dikurangi 15 menit. Sehingga akan diperoleh data open price 15 menit yang lalu. Lalu fungsi ini akan dipanggil sebanyak ‘x’ ,’ x’ adalah jumlah periode. Berikut adalah fungsi movingavg yang memanggil openprice() sebanyak jumlah periode. Lihat gambar 5.9.

```
    public function movingAvg(){

        $periode = $this->ohlc->getPeriode(1);
        $now = time();
        $time1 = $now;
        $price= array();
        for($i=0;$i<$periode;$i++){
            $time2 = $time1-900;
            if($this->ohlc->openPrice($time1,$time2)!=null){
                $price[] = $this->ohlc->openPrice($time1,$time2);
            }
            else{
                $i--;
            }

            $time1 = $time2;
        }

        $average = array_sum($price)/count($price);
        return number_format($average,8);
    }


}
```

Gambar 5.9 Fungsi *Moving Average*

Setelah didapat hasil moving average, maka tampilkan pada kotak.

5.2.4. Menampilkan Data Order Aktif

Pada subbab ini akan dijelaskan bagaimana menampilkan data order aktif.
Untuk menampilkan data order aktif, maka dibuat sebuah fungsi getOrder untuk memperoleh list order yang masih aktif. Berikut ini adalah kode programnya: Lihat gambar 5.10.

```
public function getOrder(){
    $history = array('pair'=>'str_btc');
    $result = $this->api->btcid_query('openOrders',$history);
    $hasil;
    foreach($result as $value){
        $hasil = $value['orders'];
    }

    return $hasil;

}
```

Gambar 5.10 Fungsi getOrder

Setelah memperoleh data, tampilkan pada tabel.

5.2.5. Fitur Jual-Beli Otomatis

Pada subbab ini akan dijelaskan bagaimana cara membuat fitur jual-beli secara otomatis yang berkaitan dengan moving average.
Untuk membuat fitur tersebut pertama tama kita akan membuat fungsi cek apakah bot aktif atau tidak. Berikut adalah kode programnya: Lihat gambar 5.11.

```
184         function isActive($id){
185          $this->db->where('id_user', $id);
186          $query = $this->db->get('user');
187          if($query->num_rows()>0){
188
189                if($query->row()->is_active>0){
190                      return true;
191                }
192                else{
193                      return false;
194                }
195          }
196          else{
197                return false;
198          }
199         }
```

Gambar 5.11 Fungsi Cek Bot Aktif

Setelah itu membuat fungsi bot. Berikut adalah kode programnya: Lihat gambar 5.12.

```
public function bot(){
    $pair=$this->api->getPair(1);
    sleep(1);
    $order=$this->getOrder();
    if($this->api->isActive(1)){
        sleep(1);
        $balance = $this->api->getBalance();
        $id_order;
        $jual="sell";
        $beli="buy";
        $price_now=$this->api->getLast();
        $sell=array('pair'=>'str_btc','type'=>$jual,'price'=>$price_now['last'],'str'=>$balance['balance']['str']);
        $buy =array('pair'=>'str_btc','type'=>$beli,'price'=>$price_now['last'],'btc'=>$balance['balance']['btc']);
        if(count($order)>0){
            for($i=0; $i<count($order); $i++){
                $id_order = $order[$i]['order_id'];
                $type = $order[$i]['type'];
                $this->cancelOrder($id_order,$type);
            }}
        if(floatval($balance['balance']['str'])>0){
            sleep(1);
            if($price_now['last']<$this->movingAvg()){
            $result = $this->api->btcid_query('trade',$sell);
            }}
        if(floatval($balance['balance']['btc'])>0){
            if($price_now['last']>$this->movingAvg()){
            $result = $this->api->btcid_query('trade',$buy);
            }}
```

Gambar 5.12 Fungsi Bot

Pada fungsi Bot() terlihat alur seperti pada flowchart yang telah dibuat. Sistem akan melakukan cek apakah bot aktif atau tidak, jika tidak aktif maka tidak melakukan apa apa. Jika aktif, maka akan melakukan cek apakah ada order yang aktif atau tidak. Jika ada order aktif maka cancel order tersebut. jika tidak, maka cek apakah akun memiliki saldo atau tidak.

Jika akun memiliki saldo, maka sistem akan melakukan cek berdasarkan perbandingan dengan moving average. Bila harga saat ini lebih kecil dari moving average, sistem melakukan jual

bila memiliki stellar koin. Dan sistem akan melakukan beli jika harga sekarang lebih tinggi dari moving average.

5.2.6. Fitur Menyimpan Data Transaksi

Pada subbab ini akan dijelaskan bagaimana cara untuk memperoleh data transaksi pada *exchanger* dan kemudian data yang telah diperoleh akan disimpan ke dalam database yang telah dbuat sebelumnya.

Untuk mendapatkan data transaksi dari API server, maka dibuat fungsi getLastTrade untuk memperoleh data dari API.

Berikut ini adalah kode program untuk fungsi getLastTrade: Lihat gambar 5.13.

```
function getLastTrade(){


$vip_btc_idr_depth="https://vip.bitcoin.co.id/api/str_btc/trades";
$getcontent = json_decode(file_get_contents($vip_btc_idr_depth), true) ;

return $getcontent;

}
```

Gambar 5.13 Fungsi getLastTrade

Setelah berhasil memperoleh data, maka akan dibuat fungsi getAllTrade untuk menyimpan data ke dalam database. Berikut ini adalah kode program dari fungsi getAllTrade: Lihat gambar 5.14.

```
19          public function getAllTrade(){
20
21          $datatrade=$this->api->getLastTrade();
22          for($i=0;$i<count($datatrade);$i++){
23              if($this->api->check_id_last_trade($datatrade[$i]['tid'])){
24                  break;
25
26                  }
27              else{
28                  $data=$this->api->insert_last_trade($datatrade[$i]);
29                  |
30                  }
31              }
32          }
```

Gambar 5.14 Fungsi getAllTrade

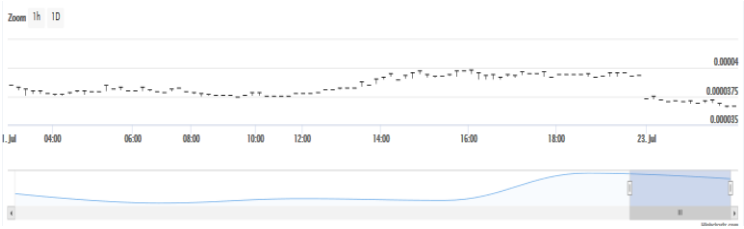
Pada fungsi getAllTrade, akan dipanggil metode getLastTrade yang berfungsi untuk memperoleh data dari API server.

Setelah data diperoleh, maka sistem akan melakukan cek pada database apakah data yang diperoleh sudah ada atau belum. Apabila data transaksi yang di ambil ternyata sudah ada di dalam database, maka akan dilewati. Jika data yang diperoleh belum ada di dalam database, maka data tersebut akan disimpan ke dalam database.

5.2.7. Menampilkan Grafik Harga

Agar mempermudah untuk melihat perubahan harga yang sebelumnya, maka perlu adanya grafik yang menampilkan history perubahan harga sebelumnya.

Berikut ini adalah gambar grafik yang telah dibuat menggunakan library javascript highchart: Lihat gambar 5.15



Gambar 5.15 Grafik Harga

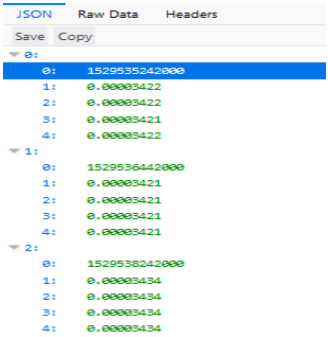
Untuk membuat grafik tersebut, maka dibuat sebuah fungsi untuk mengubah data yang ada di dalam database menjadi data yang dapat digunakan untuk grafik.

Data grafik memiliki susunan data [time, openPrice, highPrice, lowPrice, closePrice] dan data tersebut ditampilkan sebagai json. Berikut ini adalah kode program untuk merubah format data dari database menjadi format json: Lihat gambar 5.16

```php
public function dataGrafik(){
    $time1 = time();
    $time2; $arr;
    $batas = $this->api->waktuakhir();
    $sekarang = $time1;
    while($batas<$sekarang){
        $time2 = $batas;
        $time1 = $batas+600;
        if ($this->ohlc->openPrice($time1,$time2)>0){
            $sementara= array(
            $batas*1000,
            floatval($this->ohlc->openPrice($time1,$time2)),
            floatval($this->ohlc->highPrice($time1,$time2)),
            floatval($this->ohlc->lowPrice($time1,$time2)),
            floatval($this->ohlc->closePrice($time1,$time2))
            );
        $arr[] = $sementara;
        }
        $batas = $batas+600;
    }
    header('Content-Type: application/json; charset=utf-8"');
    $rt = json_encode(array_values($arr));
    echo $rt;
```

Gambar 5.16 Fungsi dataGrafik

Berikut ini adalah hasil dari data yang sudah diubah ke dalam format json. Lihat gambar 5.17



Gambar 5.17 Json dataGrafik

Selanjutnya ambil data tersebut menggunakan javascript. Berikut ini adalah kode program: Lihat gambar 5.18

```
$.getJSON('https://bitflip-feddyap.c9users.io/bot/dashboard/dataGrafik', function (data) {

    // create the chart
    Highcharts.stockChart('grafik', {
  plotOptions: {
                series: {
                    cursor: 'pointer',
                    point: {
                        events: {
                            click: function () {
  alert('Buy pada Harga '+Highcharts.dateFormat('%Y-%m-%d',this.x)+  'value: ' + this.harga);

                            }
                        }
                    }
                }
            },

        title: {
            text: 'STR/BTC'
        },
```

Gambar 5.18 Javacript Menampilkan Highchart

BAB VI
HASIL DAN PEMBAHASAN

Bab ini akan menjelaskan mengenai hasil yang didapatkan dan pembahasannya secara keseluruhan.

6.1. Hasil Jual Beli Menggunakan Metode Moving Average

Hasil yang dicapai oleh sistem jual beli secara otomatis kurang memuaskan. Menggunakan metode ini untuk jual-beli jangka pendek sangat merugikan. Berikut adalah hasil dari jual beli secara otomatis menggunakan metode moving average.

Jenis	Harga	XLM	BTC
Jual	0,00003430	102,58110465	0,00351853
Beli	0,00003440	102,58110465	0,00352879
Jual	0,00003417	15,73065861	0,00053751
Jual	0,00003417	87,54111793	0,00299128
Beli	0,00003440	4,85901162	0,00016715
Beli	0,00003435	15,85959339	0,00054477

Gambar 6.1 Hasil Jual Beli Otomatis

Sistem mengalami kerugian yang sangat banyak apabila kondisi nilai dari Moving Average berada di tengah-tengah. Contoh: Moving Average pada nilai 3435. Dan harga saat ini 3440. Maka menurut metode Moving Average, sistem akan membeli pada harga 3440, yang mengindikasikan bahwa ada tren naik. Tetapi harga jual terendah adalah 3430.

Ketika ada seseorang menjual dengan harga tersebut, maka harga saat ini adalah 3430 yang mana lebih rendah dari nilai moving average 3435. Maka sistem menganggap tren turun, dan

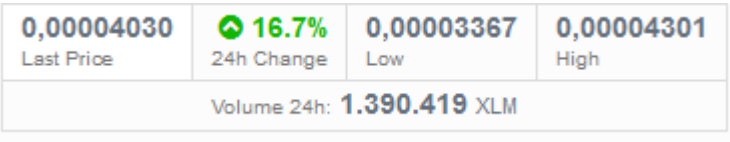
menjual semua koin yang baru dibeli. Sistem mengalami kerugian 10 poin.

6.2. Hasil Implementasi Bot

Setelah aplikasi dijalankan, maka akan diperoleh hasil sebagai berikut:

1. Kesesuaian Harga.

Gambar 6.2 Tampilan Harga Pada Aplikasi



Gambar 6.3 Tampilan Harga Pada Bitcoin.co.id

Aplikasi telah berhasil menaampilkan data yang sama dengan yang ada pada bitcoin.co.id

2. Jual Beli secara otomatis

15-Jul-18 23:19	Jual	0,00003430	102,58110465	0,00351853
15-Jul-18 21:26	Beli	0,00003440	102,58110465	0,00352879
15-Jul-18 21:23	Jual	0,00003417	15,73065861	0,00053751
15-Jul-18 21:23	Jual	0,00003417	87,54111793	0,00299128
15-Jul-18 20:44	Beli	0,00003440	4,85901162	0,00016715
15-Jul-18 20:43	Beli	0,00003435	15,85959339	0,00054477

Gambar 6.4 Riwayat Jual Beli

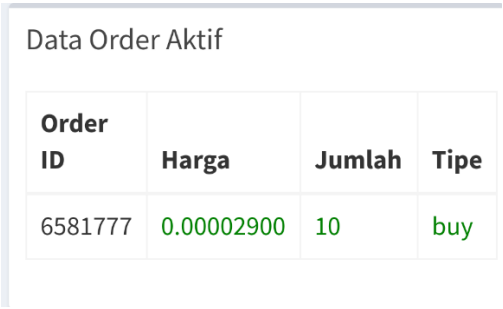
Sistem berhasil melakukan jual beli secara otomatis.

3. Menampilkan balance

Gambar 6.5 Menampilkan Saldo

Sistem berhasil menampilkan Saldo.

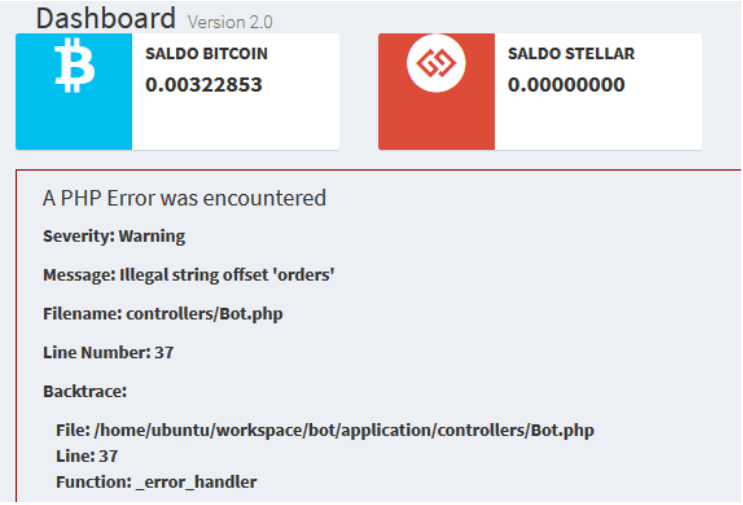
4. Menampilkan OrderAktif



Gambar 6.6 Tampilan Order Pada Aplikasi

Gambar 6.7 Tampilan Order Pada Bitcoin.co.id

5. Sebagian data tidak masuk



Gambar 6.8 Error Variabel Orders

Hal ini terjadi karena ketika sistem melakukan foreach, namun data array yang diperoleh dari API adalah kemungkinan error handle. Array orders dari API yang akan di foreach tidak ada.

6. Menggunakan Sleep(1)

   Ketika melakukan request ke API server pada fungsi bot() , fungsi lain juga melakukan hal yang sama pada satu waktu tersebut sehingga pada fungsi bot terjadi *error* karena *nonce* yang dikirim oleh fungsi bot() waktunya sama dengan fungsi lain. Hal ini disebabkan karena API server pada bitcoin,co,id hanya memperbolehkan 3 *request* dalam satu detik. Karena keterbatasan itulah, maka menunda 1 detik dalam mengirim *request* itu diperlukan. Lihat pada gambar 5.12.

“Halaman ini sengaja dikosongkan”

BAB VII
KESIMPULAN DAN SARAN

Pada bab ini akan dijelaskan mengenai kesimpulan dari hasil penelitian pada pengerjaan tugas akhir dan saran perbaikan untuk penelitian selanjutnya.

7.1. Kesimpulan

Berdasarkan hasil penelitian didapatkan beberapa simpulan yang dijelaskan ke dalam beberapa poin berikut ini:

1. Penelitian ini membuktikan bahwa akuisisi data transaksi jual beli pada API server bitcoin.co.id berhasil dilakukan. Data mengenai informasi saldo, informasi harga saat ini, informasi transaksi jual beli berhasil diperoleh.
2. Sistem berhasil melakukan jual beli secara otomatis dengan menggunakan metode *moving average* selama satu bulan.
3. Sistem telah berhasil melakukan perhitungan harga moving average dan menampilkannya.
4. Penelitian ini membuktikan bahwa setelah aplikasi berjalan selama satu bulan, sistem yang menggunakan metode *moving average* mengalami kerugian.
5. Selama aplikasi dijalankan, Aplikasi mengalami berbagai *error* dikarenakan tidak adanya sebagian data yang diperoleh dari API server bitcoin.co.id. Penyebabnya adalah karena API server bitcoin.co.id membatasi hanya 3 request per detik dan respon API server sangat lambat ketika *market* sedang ramai.

7.2. Saran

Adapun saran yang dapat disampaikan penulis untuk penelitian selanjutnya:

1. Sistem dikembangkan dengan algoritma *exponential moving average* yang merupakan versi perbaikan

untuk metode *simple moving average* untuk memperbaiki pengambilan keputusan oleh sistem.

2. Menambahkan fungsi untuk mengatur kerugian maksimal yang bisa diterima.
3. Menambah waktu interval setiap 10 detik menjadi 15 detik.
4. Mencoba *exchanger* lain yang memiliki respon *server* yang lebih baik daripada bitcoin.co.id.

“Halaman ini sengaja dikosongkan”

DAFTAR PUSTAKA

[1] S. Nakamoto, "Bitcoin: Peer-to-Peer Electronic Cash System," 2008.

[2] D. Chaum, "Blind Signature For Untraceable Payments," 1983.

[3] A. T. Sander Thomas, "On Anonymous Electronic Cash and Crime," 1999.

[4] M. S. R. L. Anthony Hadi, "Pembuatan Market Expert Advisor pada Currency Market menggunakan Fibonacci, Stochastic dan MACD Indicator," 2013.

[5] I. H. R. A. Muhammad Abdillah, "Perancangan Dan Implementasi Prosesor Scrypt Untuk Cryptocurrency Dengan Arsitektur Pipeline Berbasis Fpga," 2015.

[6] J. McCaleb, "Stellar," [Online]. Available: https://www.stellar.org. [Accessed 24 Maret 2018].

[7] J. McCaleb, "Bitcointalk," 06 April 2016. [Online]. Available: https://bitcointalk.org/index.php?topic=1428573.0. [Accessed 24 Maret 2018].

[8] M. C. Tom, "The Technical Analysis Method of Moving Average Trading: Rules That Reduce the Number of Losing Trade," 2011.

BIODATA PENULIS

Penulis bernama lengkap Feddy Anugerah Pratama. Lahir di Surabaya, tanggal 31 Desember 1993. Penulis telah menempuh pendidikan formal di SD Negeri 3 Surabaya, SMP Negeri 3 Surabaya, serta SMA Negeri 6 Surabaya. Setelah tamat pendidikan Sekolah Menengah Atas, penulis melanjutkan studi Perguruan Tinggi di Institut Teknologi Sepuluh Nopember Surabaya, diterima di Departemen Sistem Informasi dengan NRP 5211100028. Penulis memiliki hobi membaca cerita.

Pada pengerjaan Tugas Akhir di Jurusan Sistem Informasi ITS, penulis mengambil bidang minat Akuisisi Data dan Diseminasi Informasi. Penulis dapat dihubungi melalui e-mail feddyanugerah@gmail.com.

“Halaman ini sengaja dikosongkan”